Fast Correlation Computation Method for Matching Pursuit Algorithms in Compressed Sensing

Kee-Hoon Kim, Hosung Park, Seokbeom Hong, Jong-Seon No, and Habong Chung

Abstract

There have been many matching pursuit algorithms (MPAs) which handle the sparse signal recovery problem a.k.a. compressed sensing (CS). In the MPAs, the correlation computation step has a dominant computational complexity. In this letter, we propose a new fast correlation computation method when we use some classes of partial unitary matrices as the sensing matrix. Those partial unitary matrices include partial Fourier matrices and partial Hadamard matrices which are popular sensing matrices. The proposed correlation computation method can be applied to almost all MPAs without causing any degradation of their recovery performance. And, for most practical parameters, the proposed method can reduce the computational complexity of the MPAs substantially.

Index Terms

compressed sensing (CS), fast correlation computation, Fourier matrix, Hadamard matrix, matching pursuit algorithm (MPA).

I. Introduction

Compressed sensing (CS) is a novel sampling technique, where one can recover sparse signals from the undersampled measurements [1]. In a typical CS problem, the goal is to exactly reconstruct the $N \times 1$

K.-H. Kim, H. Park, S. Hong, and J.-S. No are with the Department of Electrical Engineering and Computer Science, INMC, Seoul National University, Seoul, 151-744, Korea (phone: +82-2-880-8437, fax: +82-2-880-8222, email: kkh@ccl.snu.ac.kr, lovepk98@snu.ac.kr, fousbyus@ccl.snu.ac.kr, jsno@snu.ac.kr).

H. Chung is with the School of Electronics and Electrical Engineering, Hong-Ik University, Seoul 121-791, Korea (e-mail: habchung@hongik.ac.kr).

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012-0000186).

K-sparse signal vector x based on the $M \times 1$ *measurement vector* y. By K-sparse we mean that there are at most K nonzero elements in x. The vectors x and y are linearly related to each other as

$$y = \Phi x + \eta, \tag{1}$$

where Φ is the $M \times N$ *sensing matrix* and η is the $M \times 1$ noise vector. And the relation of K, M, and N is generally $K < M \ll N$.

For the sensing matrix Φ, partial Fourier matrices and partial Hadamard matrices are popular sensing matrices, where we mean that the partial matrix is constructed by some M rows of $N \times N$ original matrix A. In other words, $\Phi = S_\Omega A$, where S_Ω is the $M \times N$ row selection matrix consisting of M rows (indices from some index set Ω) of $N \times N$ identity matrix I.

Firstly, the partial Fourier matrix is frequently used because of its good recovery performance, fast implementation using the fast Fourier transform (FFT), and applicability to practical signals. The examples include channel estimation in communication systems [2] and magnetic resonance imaging (MRI) [3]. For the partial Fourier matrix, the index set Ω can be constructed randomly or based on the cyclic difference set [4].

Secondly, some recent researches showed that well-designed deterministic sensing matrices based on linear block codes have better performance and less complexity for signal recovery compared to random sensing matrices [5], [6]. It is well known that a sensing matrix whose columns are bipolar-presented codewords of a binary linear block code can be viewed as a partial Hadamard matrix. And we can exploit the efficiency of the fast Hadamard transform (FHT).

To recover x in (1), matching pursuit algorithms (MPAs) find a sparse estimation of the signal x from y in a greedy fashion. It works iteratively by choosing the component that has the highest correlation with the current residual. Examples include the orthogonal matching pursuit (OMP) [7] and its modified versions such as the compressive sampling matching pursuit (CoSaMP) [8], the regularized OMP (ROMP) [9], the subspace pursuit (SP) [10], and the backtracking-based matching pursuit (BB MP) [11]. For instance, we summarize the OMP which is the most basic algorithm among the MPAs. The steps marked by $\lozenge$ are the common steps to the MPAs.

Algorithm 1.1 Conventional OMP recovery algorithm

---

1) Initialize : $r_0 = y$, $\Lambda_0 = \emptyset$, $t = 1$. $\lozenge$

2) Correlation computation : $h_{t-1} = \Phi^H r_{t-1}$. $\lozenge$

3) Identification : $\lambda_t = \arg\max_{j=1,\dots,N} |h_{t-1}(j)|$.

4) Augment the index set : $\Lambda_t = \Lambda_{t-1} \cup \{\lambda_t\}$.

5) Construct Φ_t : $\Phi_t = \Phi S_{\Lambda_t}^T$. $\Diamond$

6) Least squares : $x_t = (\Phi_t^H \Phi_t)^{-1} \Phi_t^H y$.

7) Update current residual : $a_t = \Phi_t x_t$, $r_t = y - a_t$. $\Diamond$

8) $t = t + 1$, return to 2) if the halting criterion is not triggered. $\Diamond$

In Algorithm 1.1, performing $h_{t-1} = \Phi^H r_{t-1}$ in 2) can be viewed as computing the correlations between the current residual r_{t-1} and the columns of Φ. And we denote h_{t-1} as the correlation vector at the t-th iteration. The whole computational complexity of the OMP is dominated by the correlation computation step and so are the other MPAs'.

In this letter, we propose a new fast correlation computation method which can be applied to almost all MPAs including OMP, CoSaMP, ROMP, SP, and BB MP. The recovery performances of the MPAs applied by the proposed method are exactly the same as those of the original MPAs. And, for most practical parameters, the proposed method can reduce the computational complexity of the MPAs substantially. The proposed method can operate only when the sensing matrix is the partial unitary matrix satisfying the following two constraints :

1) Every element of the unitary matrix U has the magnitude $1/\sqrt{N}$.
2) The set $\{\sqrt{N}u_1, \sqrt{N}u_2, \cdots, \sqrt{N}u_N\}$, where u_n is the n-th column of U, is closed under element-wise multiplication $\circ$.

At a glance, the above constraints seem to be too strict, however, the Fourier matrix and the Hadamard matrix are two kinds of the unitary matrices with these constraints. Therefore, the proposed method is meaningful and it can be widely adopted in CS.

II. A New Fast Correlation Computation Method for MPAs

In this section, we describe the proposed fast correlation computation method for general MPAs. The MPAs have the common steps marked by $\Diamond$ in Algorithm 1.1 and we derive the fast correlation computation method based on only those steps. In the following derivation, U is the unitary matrix satisfying the two constraints and the sensing matrix is $\Phi = S_\Omega U$. For simplicity, we handle not the t-th iteration but the $(t+1)$-th iteration.

Using the steps 5) and 7) in Algorithm 1.1, the correlation computation step 2) at the $(t+1)$-th iteration $h_t = \Phi^H r_t$ can be rewritten as

$$\begin{aligned} h_t =& U^H S_\Omega^T r_t \\ =& U^H S_\Omega^T (y - a_t) \\ =& U^H S_\Omega^T y - U^H S_\Omega^T a_t \\ =& h_0 - U^H S_\Omega^T S_\Omega U S_{\Lambda_t}^T x_t, \end{aligned} \tag{2}$$

where $h_0 = \Phi^H r_0 = U^H S_\Omega^T y$.

In (2), $S_{\Lambda_t}^T x_t$ can be represented as

$$S_{\Lambda_t}^T x_t = \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) e_{\Lambda_t(\tau)}, \tag{3}$$

where $e_{\Lambda_t(\tau)}$ is the $\Lambda_t(\tau)$-th column of I, $|\Lambda_t|$ is the cardinality of the index set Λ_t, and $x_t(\tau)$ is the τ-th element of x_t. By using (2) and (3), we obtain

$$\begin{aligned} h_t =& h_0 - U^H S_\Omega^T S_\Omega U \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) e_{\Lambda_t(\tau)} \\ =& h_0 - \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) U^H S_\Omega^T S_\Omega U e_{\Lambda_t(\tau)}. \end{aligned} \tag{4}$$

Without loss of generality, we assume the first column of U is $(1/\sqrt{N}, 1/\sqrt{N}, \cdots, 1\sqrt{N})^T$. And (4) can be rewritten as

$$h_t = h_0 - \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) U^H S_\Omega^T S_\Omega D_{\Lambda_t(\tau)} U e_1, \tag{5}$$

where $D_{\Lambda_t(\tau)} = \sqrt{N} \cdot \mathrm{diag}(u_{\Lambda_t(\tau)})$ and $u_{\Lambda_t(\tau)}$ is the $\Lambda_t(\tau)$-th column of U. Because the matrix $S_\Omega^T S_\Omega$ is the diagonal matrix, $S_\Omega^T S_\Omega D_{\Lambda_t(\tau)} = D_{\Lambda_t(\tau)} S_\Omega^T S_\Omega$ and (5) can be rewritten as

$$h_t = h_0 - \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) U^H D_{\Lambda_t(\tau)} S_\Omega^T S_\Omega U e_1. \tag{6}$$

We denote $P_{\Lambda_t(\tau)} = U^H D_{\Lambda_t(\tau)} U$ and thus $P_{\Lambda_t(\tau)} U^H = U^H D_{\Lambda_t(\tau)}$. Consequently, the correlation computation at the $(t+1)$-th iteration can be expressed as

$$\begin{aligned} h_t =& h_0 - \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) P_{\Lambda_t(\tau)} U^H S_\Omega^T S_\Omega U e_1 \\ =& h_0 - \sum_{\tau=1}^{|\Lambda_t|} x_t(\tau) P_{\Lambda_t(\tau)} c, \end{aligned} \tag{7}$$

where $c = U^H S_\Omega^T S_\Omega U e_1$ which is called the *correlation kernel vector*. Note that the correlation kernel vector c is independent to the sparse signal vector x and thus can be stored in advance. The matrix $P_{\Lambda_t(\tau)}$ in (7) is a permutation matrix according to the following theorem. The permutation matrix can be performed with negligible computational complexity because of its structure.

Theorem 2-1 : $P_{\Lambda_t(\tau)} = U^H D_{\Lambda_t(\tau)} U$ is a permutation matrix (i.e., a square binary matrix that has exactly one element 1 in each row and each column and 0s elsewhere) if the unitary matrix U is under the two constraints.

Proof of Theorem 2-1 : $U^H D_{\Lambda_t(\tau)} U$ can be expressed as

$$\begin{aligned} U^H D_{\Lambda_t(\tau)} U =& (u_1 \ \cdots \ u_N)^H \cdot \sqrt{N} \cdot \mathrm{diag}(u_{\Lambda_t(\tau)}) \cdot (u_1 \ \cdots \ u_N) \\ =& \frac{1}{\sqrt{N}} \cdot (u_1 \ \cdots \ u_N)^H \cdot \left(\sqrt{N} u_{\Lambda_t(\tau)} \circ \sqrt{N} u_1 \ \cdots \ \sqrt{N} u_{\Lambda_t(\tau)} \circ \sqrt{N} u_N \right). \end{aligned} \tag{8}$$

$\sqrt{N} u_{\Lambda_t(\tau)} \circ \sqrt{N} u_n$, $n = 1, \cdots, N$, are distinct column vectors because their elements are nonzero by the first constraint of U. And each vector belongs to the set $\{\sqrt{N}u_1, \sqrt{N}u_2, \cdots, \sqrt{N}u_N\}$ because of the second constraint of U. Therefore, (8) can be rewritten as

$$\begin{aligned} U^H D_{\Lambda_t(\tau)} U =& \frac{1}{\sqrt{N}} \cdot (u_1 \ \cdots \ u_N)^H \cdot \left(\sqrt{N} u_1 \ \cdots \ \sqrt{N} u_N \right) \cdot P_{\Lambda_t(\tau)} \\ =& (u_1 \ \cdots \ u_N)^H \cdot (u_1 \ \cdots \ u_N) \cdot P_{\Lambda_t(\tau)} \\ =& I P_{\Lambda_t(\tau)} = P_{\Lambda_t(\tau)}, \end{aligned} \tag{9}$$

where $P_{\Lambda_t(\tau)}$ is the permutation matrix which is determined by $\Lambda_t(\tau)$ and the structure of U. □

To sum it up, the correlation computation at the t-th iteration (i.e., computing h_{t-1}) can be performed by $|\Lambda_{t-1}|$ subtractions of properly scaled and permutated versions of the correlation kernel vector c to the initial correlation vector h_0.

III. Fast OMP Recovery Algorithm

In this section, we apply the fast correlation computation method to the conventional OMP. And we discuss the complexity of the proposed OMP algorithm applied by the proposed method. Applying the proposed correlation computation method to other MPAs is straightforward and entirely analogous with this section.

A. Fast Correlation Computation for the OMP

The proposed correlation computation method (7) for a general MPA can be easily converted for the OMP as

$$h_{t-1} = \begin{cases} \Phi^H r_0, & t = 1 \\ h_0 - \sum_{\tau=1}^{t-1} x_{t-1}(\tau) P_{\lambda_\tau} c, & t > 1. \end{cases} \tag{10}$$

And the proposed OMP recovery algorithm can be given by simply replacing the correlation computation step 2) in Algorithm 1.1 with (10).

Note that the proposed OMP algorithm is actually identical to the conventional OMP algorithm. The only difference is the computation method and thus the proposed OMP guarantees the same recovery performance compared to the conventional OMP.

B. Complexity Analysis

In this subsection, we investigate the computational complexity of the proposed OMP algorithm in the cases of using the partial Fourier matrix and the partial Hadamard matrix. Firstly, for each matrix, we will discuss the properness of the proposed algorithm in terms of the storage requirements for the permutation matrices. Secondly, we compare the computational complexity of the proposed OMP algorithm to that of the conventional OMP algorithm. For the exact comparison, we consider the number of flops of each algorithm. And we regard one complex multiplication as 6 flops and one complex addition as 2 flops.

We remark that the proposed OMP performs the $(t-1)$ subtractions of properly scaled and permutated versions of the correlation kernel vector at the t-th iteration to compute the correlation vector in the second equation in (10). We consider the case when N is a power of two, which is used very often in signal processing. But, the proposed method can be used for any N.

1) Storage Requirements Using the Partial Fourier Matrix: When we use the partial Fourier matrix as the sensing matrix, from the discrete Fourier transform (DFT) properties, P_{λ_τ} is the matrix which cyclically shifts c when P_{λ_τ} is multiplied with the vector c from the left. Therefore, the storage requirements for the permutation matrices can be negligible.

2) Computational Complexity Using the Partial Fourier Matrix: It is well known that the correlation computation at each iteration in the conventional OMP can be implemented by the N-point FFT. The N-point FFT requires $5N \log_2 N$ flops.

For the proposed OMP, in (10), the first iteration ($t = 1$) can be implemented by one FFT. And, when $t > 1$, the correlation vector is computed by using the correlation kernel vector. Exploiting the conjugate

TABLE I

COMPARISON BETWEEN THE PROPOSED OMP AND THE CONVENTIONAL OMP (# OF FLOPS AT THE t-TH ITERATION)

Φ : Partial Fourier matrix		
t	$=1$	>1
Conventional OMP	$5N\log_2 N$	$5N\log_2 N$
Proposed OMP	$5N\log_2 N$	$6N(t-1)$

Φ : Partial Hadamard matrix		
t	$=1$	>1
Conventional OMP	$2N\log_2 N$	$2N\log_2 N$
Proposed OMP	$2N\log_2 N$	$2N(t-1)$

symmetric property of the correlation kernel vector using the partial Fourier matrix as the sensing matrix, performing the second equation in (10) has the cost of $(N/2)(t-1)$ complex multiplications, $2(N/2)(t-1)$ real additions, and $N(t-1)$ complex additions at the t-th iteration. Aggregately, the proposed OMP requires $6N(t-1)$ flops at the t-th iteration.

3) Storage Requirements Using the Partial Hadamard Matrix: The storage requirements of the proposed OMP algorithm with partial Hadamard matrix are also favorable. There is no need to store the entire N permutation matrices. If we store only the $\log_2 N$ permutation matrices, the permutation matrix P_{λ_τ} for any λ_τ can be easily performed by sequentially applying some matrices among the stored $\log_2 N$ permutation matrices. It is easily induced from the properties of the Hadamard matrix.

4) Computational Complexity Using the Partial Hadamard Matrix: It is well known that the correlation computation at each iteration in the conventional OMP can be implemented by the N-point FHT. The N-point FHT requires $N\log_2 N$ complex additions (i.e., $2N\log_2 N$ flops).

For the proposed OMP, in (10), the first iteration ($t=1$) can be implemented by one FHT. And, when $t>1$, the correlation vector is computed by using the correlation kernel vector. Because the correlation kernel vector consists of only a small number of values compared to N using the partial Hadamard matrix as the sensing matrix, performing the second equation in (10) has approximately the cost of $N(t-1)$ complex additions. Table I summarizes this subsection.

IV. NUMERICAL ANALYSIS

Here we present some numerical results characterizing the performance of the proposed OMP algorithm compared to the conventional OMP algorithm. The results were produced using the partial Fourier matrices

and the partial Hadamard matrices with practical and various sizes. And we plot the computational complexities for $1 \leq t \leq 13$, which is reasonable for given M and N.

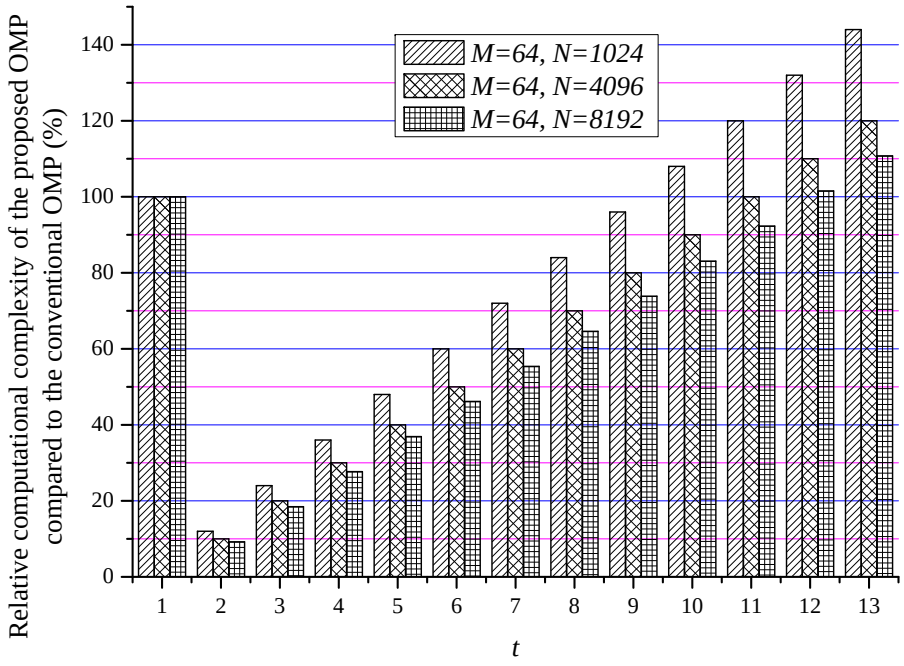

Fig. 1. Relative computational complexity of the proposed OMP compared to the conventional OMP at the t-th iteration when the partial Fourier matrices are used.

Fig. 1 shows the relative computational complexity of the proposed OMP compared to the conventional OMP when the partial Fourier matrices are used. Because the computational complexity of the proposed OMP algorithm at the t-th iteration is proportional to $t-1$, there is a excessive point and thus adaptive strategy is needed. For instance, for $M = 64$ and $N = 4096$, $t = 11$ is the excessive point and the conventional OMP can be used from the 12-th iteration. Consequently, the proposed OMP algorithm has a benefit to reduce the computational complexity substantially. Especially, for large N, the proposed OMP has a good benefit.

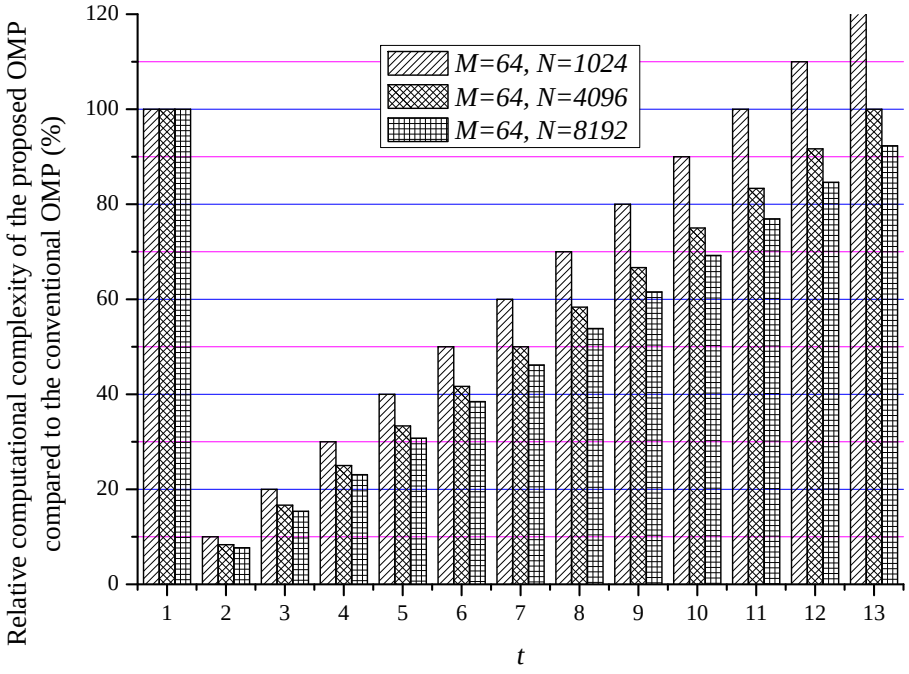

Fig. 2. Relative computational complexity of the proposed OMP compared to the conventional OMP at the t-th iteration when the partial Hadamard matrices are used.

Fig. 2 shows the relative computational complexity of the proposed OMP compared to the conventional OMP when the partial Hadamard matrices are used. Like the case of using the partial Fourier matrices in Fig. 1, the proposed OMP algorithm has a benefit to reduce the computational complexity substantially.

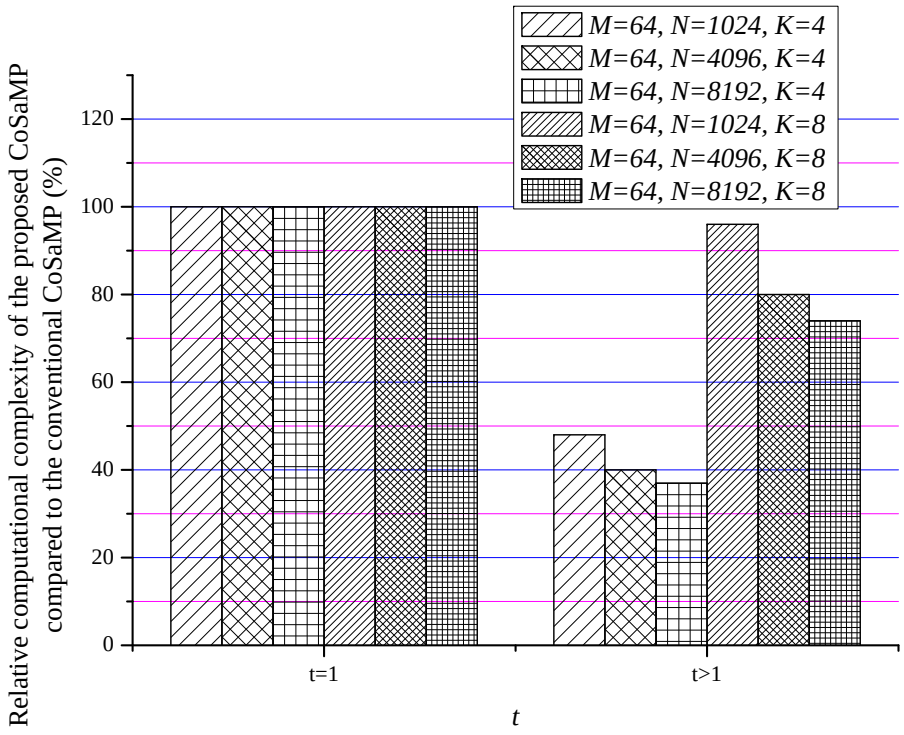

Fig. 3. Relative computational complexity of the proposed CoSaMP compared to the conventional CoSaMP at the t-th iteration when the partial Fourier matrices are used.

Besides the proposed OMP, we present the numerical result when the proposed correlation computation method is applied to the CoSaMP [8]. Due to lack of space, we leave out the detailed description of the proposed CoSaMP. Fig. 3 shows the relative computational complexity of the proposed CoSaMP compared to the conventional CoSaMP at the t-th iteration. We use the partial Fourier matrices as the sensing matrix. Different to the proposed OMP, the proposed CoSaMP requires the same computational cost at each iteration except when $t = 1$. Especially, the proposed CoSaMP algorithm has a good benefit for small K and large N. For instance, when $M = 64$, $N = 8192$, and $K = 4$, the proposed CoSaMP requires only the 37% computational cost compared to the conventional CoSaMP.

REFERENCES

[1] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.

[2] W. U. Bajwa, J. Haupt, A. M. Sayeed, and R. Nowak, "Compressed channel sensing: A new approach to estimating sparse multipath channels," *Proc. IEEE*, vol. 98, pp. 1058–1076, Jun. 2010.

[3] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, "Compressed sensing MRI," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 72–82, Mar. 2008.

[4] N. Y. Yu, "Deterministic construction of partial Fourier compressed sensing matrices via cyclic difference sets," *arXiv:1008.0885v1 [cs.IT]*, Aug. 2010.

[5] R. Calderbank, S. Howard, and S. Jafarpour, "Construction of a large class of deterministic sensing matrices that satisfy a statistical isometry property," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 2, pp. 358–374, Apr. 2010.

[6] S. Hong, H. Park, B. Shin, J.-S. No, and H. Chung, "A new performance measure using k-set correlation for compressed sensing matrices," *IEEE Signal Process. Lett.*, vol. 19, no. 3, pp. 143–146, Mar. 2012.

[7] J. A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[8] D. Needell and J. A. Tropp, "CoSaMP: Iterativesignal recovery from incomplete and inaccurate samples," *Appl. Comput. Harmon. Anal.*, vol. 26, no. 3, pp. 301–321, May 2009.

[9] D. Needell and R. Vershynin, "Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit," *IEEE J. Select. Top. Signal Process.*, vol. 4, no. 2, pp. 310–316, Apr. 2010.

[10] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2230–2249, Apr. 2010.

[11] H. Huang and A. Makur, "Backtracking-based matching pursuit method for sparse signal reconstruction," *IEEE Signal Process. Lett.*, vol. 18, no. 7, pp. 391–394, Jul. 2011.